Old Horizon

Ref: Old Horizon V1
Author: Gareth I Jenkins
Date: [SAVEDATE * MERGEFORMAT]

1. Introduction

This paper was prepared by Gareth Jenkins and addresses two questions about the old Horizon system:

1. Injection of Data by Support Staff
2. Protection against Corruption of Data when transmitted between the Branch and Data Centre (and vice versa)

The old Horizon system was based on the Riposte product developed by the Escher Corporation in USA. Most of the features described in this note are provided by the Riposte system. Before answering the questions directly a brief overview of Riposte is provided as background.

2. Overview of Riposte

Riposte is primarily an asynchronous messaging system.

Please be aware that this overview omits the detail where possible to aid a simple description. I have included notes in boxes such as this to indicate added complexities that may be relevant.

I have previously produced a note describing the overall integrity of Riposte which I have used as part of my standard Witness Statements. Some of the material in that note may also be relevant here. I don't have that information at home, but could perhaps retrieve a reference to it and the note itself if I visited BRA01.

It has the following key features:

1. Each message is uniquely identified with 3 key attributes:
    a. The Branch with which it is associated
    b. The Node (ie the Counter or pseudo counter when messages are generated in the Data Centre) at which it was originally created.
    c. A unique, monotonically increasing sequence number associated with the Branch and Node that ensures that the message is unique and also ensures that any missing messages can be clearly highlighted
2. Once a message has been generated, it is not possible to modify that message in any way. However it is possible to write a similar message that counteracts the original message, whilst the original message still exists (for example when the price changes for an item in Reference Data e.g. for a first class stamp).

3. If the message is generated when a User is Logged On at a Counter, then the User Id of the Logged On User is included in the message.
4. Each Message includes a CRC (Cyclic Redundancy Check) Attribute which is an 8 hexadecimal character checksum of the entire message which is checked whenever the message is copied from one Node to another to ensure that the message has not been corrupted.
5. A separate instance of Riposte runs on each Node (ie Counter within the Branches) and on 4 separate "Correspondence Servers" Nodes in the Data Centre.
6. Each instance of Riposte is configured with one or more neighbours with which it communicates.
7. Each Counter has every other Counter in the Branch configured as a neighbour.
8. Counter 1 in each Branch is defined as a "Gateway Counter" and as well as having all other counters in the Branch configured as Neighbours, it is also configured to be the neighbour of the 4 Correspondence Servers in the Data Centre.

In the Live system there were actually 16 Correspondence servers, but they were configured as 4 separate clusters of 4 Correspondence servers each covering 25% of the Post Office Branches.

In a single counter Branch a separate instance of Riposte was configured to run on a removable Hard Disk, so that should the Gateway need to be replaced, that removable Hard Disk could be transferred to the replacement Gateway, thus reducing the amount of data to be copied from the Data Centre as part of recovery.

9. When a new message was created at a Node then after it had been secured to the local hard disk, a copy was sent to all of its neighbours. This was known as "Replication". When such messages were received then they would be secured to the local hard disk as being associated with the originating Node. They were then replicated to all of the receiving Node's neighbours. In this way, data from all Nodes in a Branch would eventually be copied to all the other counters within the Branch and also to all the Correspondence Servers in the Data Centre. Similarly, Data from all Correspondence Servers would eventually be visible at all counters.

Note that replication may not be immediate. This allowed Branches to operate off line when there was no communication with the Data Centre, for example following a LAN failure within a Branch.

There were a number of features that allowed the frequency of replication to be controlled.

In many cases, a number of related messages make up a Transaction (for example a Customer's Basket) and the replication process also ensures that either an entire transaction is replicated or none of it is replicated.

10. Whenever Riposte started up on a counter, it would attempt to communicate with all of its neighbours and check the highest sequence number associated

with each Branch and Counter. If they were the same, then all was well. If, however, they were different, requests were sent to the neighbour requesting missing messages so that that instance of Riposte was fully up to date with all messages before it started normal operations.

11. At the counter, the Horizon Counter Application would write messages related to transactions that took place on the counter. Message were also used to record significant events (such as Log On / Log Off, Printing reports or receipts, making Declarations and balancing activities).

12. At the Data Centre, messages were written by software known as Agents to communicate with the counter. For example this would be used to modify Reference Data.

13. For an Online transaction (for example a Banking Authorisation), the following sequence would take place:

    a. The counter would write a message requesting the authorisation. This would be marked as a priority message to ensure immediate, replication to the Data Centre

    b. An Agent, running in the Data Centre would be notified of the arrival of this Request and communicate with the Banks to obtain Authorisation. The Response would then be written by the Agent to the Correspondence Server (again as a priority message)

    c. The counter would wait until it received the response message (or timed out) and then act on its content.

    This is again an example of a message initiated by an agent at the Correspondence Server rather than at a counter.

14. One of the Agents was an "Audit Agent", that was run on 2 of the 4 Correspondence Servers in each Cluster, that was responsible for picking up all message that were received at that Correspondence Server and copying them to a set of text files which made up the Audit trail. It is these files that are used to provide details of the transactions carried out at any Branch. The Audit trail for a Branch consists of all messages written to that Branch either at counters within the Branch or the Correspondence servers. The extract tools present this information as XML data and this is normally filtered to produce separate spreadsheets of Transactions and Events. However the raw XML data associated with a Branch is also available for Audit analysis.

The fact that there are 2 Audit agents, means that there are two independent Audit trails, which should be logically equivalent. Audit retrieval can be made from either audit trail.

3. Injection of Data

Riposte provided a number of utilities that could be used to write message to the message store. There were processes in place to allow 3rd line support staff to inject messages if necessary in order to correct issues. Any such injection would have been subject to the operational change processes and any such change would have been signed off by Post Office Ltd.

The SSC may have details of any such process. I don't have such information.

I am not aware of any specific instances where this would have happened and I do know that it would only have been done if there was no other option to correct a fault.

Note that any such message would be included in the audit trail described in section [REF _Ref423948364 \r \h] point [REF _Ref423948344 \r \h]. This means that it should be possible to identify them if they had occurred. However such identification would be time consuming for the following reasons:

1. It would be necessary to extract all the data associate with any branch that was being checked.

This has already been done for those Branches being investigated by Second Sight, but as that equates to a small number (less than 200) of the 14,000 Branches open at the time for which data is available – extending the analysis to all Branches would be very labour intensive and time consuming.

2. From the raw data extracted, it would then be necessary to filter out any "normal messages". This would consist of:
    a. Anything written at the counter
    b. Anything written by the normal agents running at the Data Centre

   This should leave nothing left to examine. However some effort would be required to ensure that the filter of messages written by "normal agents running at the Data Centre" was properly defined

3. It is expected that the Injection process would include an indication of why the message was injected and by whom and this should enable such messages to be related to a specific Operational incident.

The alternative approach is to go through the logs of Operational incidents looking for any that were resolved by injecting messages to Riposte.

4. Corruption of Data

As described in Section [REF _Ref423948364 \r \h] point [REF _Ref423949671 \r \h], each message includes a CRC attribute. This is checked whenever the message is replicated between Nodes and also whenever it is read (including when it is read by the audit retrieval software) to ensure that the message has not been corrupted. The 8 character CRC has 16^8 possible values (approx. 4 billion) and so is highly unlikely not to pick up any corruption within the data of the message.